



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/847,309	05/03/2001	Alan Hartman	IL9-2000-0079	5436

7590

05/12/2004

McGuireWoods LLP
Suite 1800
1750 Tysons Boulevard
McLean, VA 22102

EXAMINER

INGBERG, TODD D

ART UNIT	PAPER NUMBER
----------	--------------

2124

DATE MAILED: 05/12/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

09/847,309

Applicant(s)

HARTMAN ET AL.

Examiner

Todd Ingberg

Art Unit

2124

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 20 July 2001.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-81 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-81 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
 - ☐ Certified copies of the priority documents have been received in Application No. _____.
 - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date 4 / July 30, 2001.
- 4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____.
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: _____.

Art Unit: 2124

DETAILED ACTION

Claims 1 – 81 have been examined.

Information Disclosure Statement

1. The information disclosure statement filed July 30, 2001 has been considered.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

3. Claims 6, 7, 20, 21, 35, 36, 54 55, 74 and 75 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. The following limitations have drawn the rejection for being indefinite:

“ said coverage variables has **assumed all possible values** thereof. “, no concrete value is specified for **all possible values** for the coverage variables. The number of possible values of a variable is a factorial of the number of printable characters in the character set (i.e. ASCII) in combination of the maximum word length of the machine. The ability to handle this extraordinary number of values is not supported in the Specification and is deemed indefinite.

Interpretation

The Examiner is interpreting the meaning of the limitations to be similar to a user defined scalar where the configuration variable is set to have several meanings and these values are used in the testing. TETware supports this in the configuration file. Clear and concise wording would resolve this issue.

Claim Objections

4. Claim 60 is objected to under 37 CFR 1.75(c), as being of improper because it is a duplicate of 42. Examiner believes Applicant intended to have it under independent claim 47 not 29.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

6. Claims 1 – 81 are rejected under 35 U.S.C. 102(b) as being anticipated by The Open Group's product **TETware** as released September 18, 1998 and documented in the following documents.

TETware User Guide Revision 1.2 (referred to as **User**)

TETware Programmers Guide Revision 1.2 (referred to as **Prog**)

TETware Release Notes Release 3.3 (referred to as **Notes**)

The simultaneous update of this documents constitute a single reference since they document the same product.

Claim 1

TETware anticipates a method of test generation for testing computer software (TETware , **User**, page 1, Preface), comprising the steps of: modeling a software application as a finite state machine to define a behavioral model (**User**, page 26, Scenario file) ; associating said behavioral model with a focus (**User**, pages 26-30, the placing of directives in the code) , said focus having a reference to said behavioral model (as per above), and having at least one directive (as per above); and generating a test program according to state transitions of said behavioral model and said directive of said focus(**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Art Unit: 2124

Claim 2

The method according to claim 1, wherein said directive comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 3

The method according to claim 1, wherein said directive comprises a model dependent directive, and a coverage variable of said behavioral model is tagged by a tag of said model dependent directive, said coverage variable having allowable values.

Interpretation

Directives in compiler theory are processed by a pre-processor. The directives are dependent on the preprocessor. As per pages 26-30.

Claim 4

The method according to claim 3, wherein said directive further comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 5

The method according to claim 3, wherein said test program references said coverage variable, and said step of generating is performed until said coverage variable has assumed each of said allowable values. (**User**, pages 32-36, possible values for configuration file).

Claim 6

The method according to claim 5, wherein said coverage variable comprises a plurality of coverage variables, and said step of generating is performed until a cross product of said coverage variables has assumed all possible values thereof. (**User**, page 36, distributed configuration file).

Claim 7

The method according to claim 5, wherein said coverage variable comprises a plurality of coverage variables, and said step of generating is performed until an orthogonal array of said coverage variables has assumed all possible values thereof. (**User**, pages 32-36, possible values for configuration file – multiple values stored in an array).

Claim 8

The method according to claim 3, wherein said model dependent directive comprises a plurality of model dependent directives, and said coverage variable is tagged by a plurality of tags of said model dependent directives.

Interpretation

Art Unit: 2124

Directives in compiler theory are processed by a pre-processor. The directives are dependent on the preprocessor. As per pages 26-30.

Claim 9

The method according to claim 3, wherein said tag is a number-of-tests-per-value tag (TETware, **User**, page 36-37, Result Code file).

Claim 11

The method according to claim 1, wherein said directive comprises a plurality of directives that are combined to define a directive expression, wherein said step of generating is performed until said directive expression has a predetermined value. (**User**, pages 26-30, directives to expressions).

Claim 12

The method according to claim 1, wherein said step of modeling is performed by retrieving said behavioral model from a model archive. (**User**, page 41-42, save file).

Claim 13

The method according to claim 1, wherein said step of associating is performed by retrieving said focus from a focus archive. (**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 14

The method according to claim 13, further comprising the steps of comparing state variables of foci of said focus archive with state variables of said behavioral model; and responsive to comparisons resulting from said step of comparing revising selected ones of said foci. (**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 15

TETware anticipates a computer software product, comprising a computer-readable medium in which computer program instructions are stored (TETware, **User**, page 1, Preface), which instructions, when read by a computer, cause the computer to execute a method of test generation for testing computer software (TETware, **User**, page 1, Preface), the method comprising the steps of:

accepting as a first input a behavioral model of a software application, wherein said behavioral model (**User**, page 26, Scenario file) comprises a finite state machine; accepting as a second input a focus having a reference to said behavioral model-, and having at least one directive; associating said behavioral model with said focus (**User**, Directives, pages 26 – 30, combination of directives and scenarios); and generating a test program according to state transitions of said behavioral model and said directive of said focus **User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Art Unit: 2124

Claim 16

The computer software product according to claim 15, wherein said directive comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 17

The computer software product according to claim 15, wherein said directive comprises a model dependent directive, and a coverage variable of said behavioral model is tagged by a tag of said model dependent directive, said coverage variable having allowable values. (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 18

The computer software product according to claim 17, wherein said directive further comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 19

The computer software product according to claim 17, wherein said test program references said coverage variable, and said step of generating is performed until said coverage variable has assumed each of said allowable values. . (User, pages 32-36, possible values for configuration file – multiple values stored in an array).

Claim 20

The computer software product according to claim 19, wherein said coverage variable comprises a plurality of coverage variables, and said step of generating is performed until a cross product of said coverage variables has assumed all possible values thereof. . (User, pages 32-36, possible values for configuration file and User, page 36, distributed configuration file).

Claim 21

The computer software product according to claim 19, wherein said coverage variable comprises a plurality of coverage variables, and said step of generating is performed until an orthogonal array of said coverage variables has assumed all possible values thereof. (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Claim 22

Art Unit: 2124

The computer software product according to claim 17, wherein said model dependent directive comprises a plurality of model dependent directives, and said coverage variable is tagged by a plurality of tags of said model dependent directives. As per claims 16 and 18.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 23

The computer software product according to claim 17, wherein said tag is a number-of-tests-per-value tag. (TETware, **User**, page 36-37, Result Code file).

Claim 25

The computer software product according to claim 15, wherein said directive comprises a plurality of directives that are combined to define a directive expression, wherein said step of generating is performed until said directive expression has a predetermined value. (**User**, pages 26-30, directives to expressions).

Claim 26

The computer software product according to claim 15, wherein said step of modeling is performed by retrieving said behavioral model from a model archive. (**User**, page 41-42, save file).

Claim 27

The computer software product according to claim 15, wherein said step of associating is performed by retrieving said focus from a focus archive. (**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 28

The computer software product according to claim 27, further comprising the steps of comparing state variables of foci of said focus archive with state variables of said behavioral model; and responsive to comparisons resulting from said step of comparing revising selected ones of said foci. (**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 29

TETware anticipates a method of test generation for testing computer software (TETware, **User**, page 1, Preface), comprising the steps of: modeling a software application as a finite state machine to define a behavioral model(**User**, page 26, Scenario file); associating said behavioral model with a focus (**User**, Scenario with Directives pages 26-30), said focus having a reference to said behavioral model, and having at least one directive (**User**, pages 26-30, directives); deriving an abstract test suite from said behavioral model and said focus(**User**, Directives, pages 26 – 30, combination of directives and scenarios), wherein said abstract test suite complies with a test constraint that is encoded in said focus (**User**, configuration file, page 32-36); executing

Art Unit: 2124

said abstract test suite in an execution engine(**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 30

The method according to claim 29, wherein said step of executing said abstract test suite comprises the step of generating a test script from said abstract test suite; wherein said test script is executed in said execution engine. (**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 31

The method according to claim 29, wherein said step of producing said abstract test suite is performed with a testing interface. (**USER**, Directives, the ability to place Directives in the code, pages 26-30).

Claim 32

The method according to claim 31, wherein said testing interface comprises an abstract-to-concrete translation table. (**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Interpretation

The translating from the abstraction of the directives of the scenario to the actual runtime of the code. This is inherently taught in TETware.

Claim 33

The method according to claim 29, wherein said testing interface comprises a test driver, having an operator interface, and further comprising the step of: varying parameters of said test driver via said operator interface in accordance with requirements of said software application. (**User**, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 34

The method according to claim 29, wherein said directive comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 35

The method according to claim 38 wherein said coverage variable comprises a plurality of coverage variables, and said step of generating is performed until a cross product of said coverage variables has assumed all possible values thereof. . (**User**, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings” and **User**, page 36, distributed configuration file).

Claim 36

Art Unit: 2124

The method according to claim 38, wherein said coverage variable comprises a plurality of coverage variables, and said step of generating is performed until an orthogonal array of said coverage variables has assumed all possible values thereof. . (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings” stored with multiple meanings is the array).

Claim 37

The method according to claim 29, wherein said directive comprises a model dependent directive, and a coverage variable of said behavioral model is tagged by a tag of said model dependent directive, said coverage variable having allowable values. (As per above “dependent directive” and User, pages 32-36, possible values for configuration file).

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 38

The method according to claim 37, wherein said abstract test suite references said coverage variable, and said step of generating is performed until said coverage variable has assumed each of said allowable values. . (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Claim 39

The method according to claim 37, wherein said directive further comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 40

The method according to claim 37, wherein said model dependent directive comprises a plurality of model dependent directives, and said coverage variable is tagged by a plurality of tags of said model dependent directives.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 41

The method according to claim 37, wherein said tag is a number-of-tests-per-value tag. (TETware, User, page 36-37, Result Code file).

Claim 43

The method according to claim 29, wherein said directive comprises a plurality of directives that are combined to define a directive expression, wherein said step of generating is performed until

Art Unit: 2124

said directive expression has a predetermined value. (User, pages 26-30, directives to expressions).

Claim 44

The method according to claim 29, wherein said step of modeling is performed by retrieving said behavioral model from a model archive. (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 45

The method according to claim 29, wherein said step of associating is performed by retrieving said focus from a focus archive. (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 46

The method according to claim 29, further comprising the steps of comparing state variables of foci of said focus archive with state variables of said behavioral model; and responsive to comparisons resulting from said step of comparing revising selected ones of said foci. (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53 and results file page 36-37).

Claim 47

TETware anticipates a computer software product for testing computer software (TETware, User, page 1, Preface), comprising a computer-readable medium in which computer program instructions are stored (TETware, User, page 1, Preface), which instructions, when read by a computer, cause the computer to perform the steps of: associating a behavioral model of a software application (User, page 26, Scenario file) with a focus, said focus having a reference to said behavioral model (User, Scenario with Directives pages 26-30), and having at least one directive (as per above), wherein said behavioral model models a finite state machine (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53); deriving an abstract test suite from said behavioral model and said focus (Use of directives as per above), wherein said abstract test suite complies with a test constraint that is encoded in said focus (User, configuration file, pages 32-36); executing said abstract test suite in an execution engine (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 48

The computer software product according to claim 47, wherein said step: of executing said abstract test suite comprises the step of generating a test script from said abstract test suite; wherein said test script is executed in said execution engine (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 49

Art Unit: 2124

The computer software product according to claim 47, wherein said step of producing said abstract test suite is performed with a testing interface. (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 50

The computer software product according to claim 49, wherein said testing interface comprises an abstract-to-concrete translation table. . (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Interpretation

The translating from the abstraction of the directives of the scenario to the actual runtime of the code. This is inherently taught in TETware.

Claim 51

The computer software product according to claim 49, wherein said testing interface comprises a test driver, having an operator interface, and further comprising the step of: varying parameters of said test driver via said operator interface in accordance with requirements of said software application. (User, scope rules as per page 26, 5.3.2.2)

Claim 52

The computer software product according to claim 47, wherein said directive comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 53

The computer software product according to claim 29, wherein said directive comprises a model dependent directive, and a coverage variable of said behavioral model is tagged by a tag of said model dependent directive, said coverage variable having allowable values. . (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Claim 54

The computer software product according to claim 53 wherein said coverage variable comprises a plurality of coverage variables, and said step of generating is performed until a cross product of said coverage variables has assumed all possible values thereof. (User, page 36, distributed configuration file).

Claim 55

The computer software product according to claim 53, wherein said coverage variable comprises a plurality of coverage variables, and said step of generating is performed until an orthogonal array of said coverage variables has assumed all possible values thereof. (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Claim 56

Art Unit: 2124

The computer software product according to claim 53, wherein said abstract test suite references said coverage variable, and said step of generating is performed until said coverage variable has assumed each of said allowable values. . (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Claim 57

The computer software product according to claim 53, wherein said directive further comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 58

The computer software product according to claim 53, wherein said model. dependent directive comprises a plurality of model dependent directives, and said coverage variable is tagged by a plurality of tags of said model dependent directives.

Interpretation

Directives in compiler theory are processed by a pre-processor. The directives are dependent on the preprocessor. As per pages 26-30.

Claim 59

The computer software product according to claim 37, wherein said tag is a number-of-tests-per-value tag. (TETware, User, page 36-37, Result Code file).

Claim 61

The computer software product according to claim 47, wherein said directive comprises a plurality of directives that are combined to define a directive expression, wherein said step of generating is performed until said directive expression has a predetermined value. (User, pages 26-30, directives to expressions).

Claim 62

The computer software product according to claim 47, wherein said step of modeling is performed by retrieving said behavioral model from a model archive. (User, page 41-42, save file).

Claim 63

The computer software product according to claim 47, wherein said step of associating is performed by retrieving said focus from a focus archive. (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 64

The computer software product according to claim 63, further comprising the steps of comparing state variables of foci of said focus archive with state variables of said behavioral model; and responsive to comparisons resulting from said step of comparing revising selected ones of said

Art Unit: 2124

foci. (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 65

TETware anticipates a computer system for testing computer software (TETware, User, page 1, Preface), comprising: a user interface for creating a behavioral model of a software application (User, page 26, Scenario file), said behavioral model representing a finite state machine, wherein said user interface creates a focus, said focus having a reference to said behavioral model (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53), and having at least one directive (as per above); a compiler (inherent since directives are pre-processor directives as in before the compiler by definition), for converting said behavioral model into an intermediate encoding thereof (User, Directives, pages 26 – 30, combination of directives and scenarios at compile time); a test generator, accepting said intermediate encoding and said focus as input, and producing an abstract test suite; an execution engine for executing a test program of said abstract test suite (User, pages 26-30, the placing of directives in the code and the state synchronization of the test manager as per pages 49-53).

Claim 66

The system according to claim 65, wherein said execution engine produces a suite execution trace (User, journal file, page 37).

Claim 67

The system according to claim 66, further comprising an analyzer which reads said suite execution trace, wherein said execution engine accepts an output of said analyzer (User, journal file, page 37).

Claim 68

The system according to claim 65, further comprising a visualizer for visualizing an output of said execution engine. (User, Trace and Debugging Facilities and ability to set Directives up as per pages 87-89 and 26-30).

Claim 69

The system according to claim 65, wherein said execution engine further receives input from an application model interface that is created by said user interface. (User, Trace and Debugging Facilities and ability to set Directives up as per pages 87-89 and 26-30).

Claim 70

The system according to claim 65, wherein said directive comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Art Unit: 2124

Claim 71

The system according to claim 65, wherein said directive comprises a model dependent directive, and a coverage variable of said behavioral model is tagged by a tag of said model dependent directive, said coverage variable having allowable values. . (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Claim 72

The system according to claim 71, wherein said directive further comprises a model independent directive.

Interpretation

Directives are statements placed in the source code prior to a compiler. The use of Directives make the tool model independent. TETware teaches directives on pages 26 – 30.

Claim 74

The system according to claim 71, wherein said test program references said coverage variable, and said test generator operates until said coverage variable has assumed each of said allowable values. . (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Claim 74

The system according to claim 73, wherein said coverage variable comprises a plurality of coverage variables, and said execution engine executes until a cross product of said coverage variables has assumed all possible values thereof. (User, page 36, distributed configuration file).

Claim 75

The system according to claim 73, wherein said coverage variable comprises a plurality of coverage variables, and said execution engine executes until an orthogonal array of said coverage variables has assumed all possible values thereof. (User, pages 32-36, possible values for configuration file – multiple values for same variable – page 32 “meanings”).

Claim 76

The system according to claim 71, wherein said model dependent directive comprises a plurality of model dependent directives, and said coverage variable is tagged by a plurality of tags of said model dependent directives.

Interpretation

Directives in compiler theory are processed by a pre-processor. The directives are dependent on the preprocessor. As per pages 26-30.

Claim 77

The system according to claim 71, wherein said tag is a number-of-tests-per-value tag. (TETware, User, page 36-37, Result Code file).

Claim 79

Art Unit: 2124

The system according to claim 65, wherein said directive comprises a plurality of directives that are combined to define a directive expression, wherein said execution engine executes until said directive expression has a predetermined value. (User, pages 26-30, directives to expressions).

Claim 80

The system according to claim 65, further comprising a model archive that's accessed by said user interface. (User, pages 41-42, Save file).

Claim 81

The system according to claim 65, further comprising a focus archive that is accessed by said user interface. (User, Trace and Debugging Facilities and ability to set Directives up as per pages 87-89 and 26-30).

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

8. Claims 10, 24, 42, 60 and 78 are rejected under 35 U.S.C. 103(a) as being unpatentable over TETware in view of the use of HTML. TETware requires a user interface to enter the Directives and perform Trace and Debugging. But TETware does not explicitly limit the User Interface. HTML interfaces are old and well known to one of ordinary skill in the art prior to the time of invention and contain mask-value directives in the tagged language such as <HTML> tag which causes a HTML to be invoked. HTML is taught in The XML Handbook (page 10). It would have been obvious to one of ordinary skill in the art to use HTML to define the user interface (tags for mask values), because the standard language is easy to use. The claims are repeated below for the claims.

Claim 10

Art Unit: 2124

The method according to claim 3, wherein said model dependent directive is a mask-value directive.

Claim 24

The computer software product according to claim 17, wherein said model dependent directive is a mask-value directive.

Claim 42

The method according to claim 37, wherein said model dependent directive is a mask-value directive.

Claim 60

The computer software product according to claim 37, wherein said model dependent directive is a mask-value directive. As per claim 42.

Claim 78

The system according to claim 71, wherein said model dependent directive is a mask-value directive.

Correspondence Information

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to **Todd Ingberg** whose telephone number is (703) 305-9775. The examiner can normally be reached during the following hours:

Monday	Tuesday	Wednesday	Thursday	Friday
6:15 – 1:30	6:15- 3:45	6:15 – 4:45	6:15-3:45	6:15-130

This schedule began December 1, 2003 and is subject to change.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, **Kakali Chaki** can be reached on (703) 305-9662. Please, note that as of August 4, 2003 the **FAX number** changed for the organization where this application or proceeding is assigned is **(703) 872-9306**.

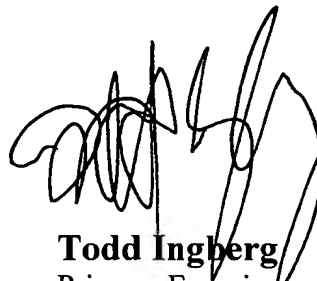
Also, be advised the United States Patent Office **new address** is

Art Unit: 2124

Post Office Box 1450

Alexandria, Virginia 22313-1450

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-9700.

A handwritten signature in black ink, appearing to read 'Todd Ingberg', with a long, sweeping line extending from the end of the signature towards the top right of the page.

Todd Ingberg
Primary Examiner
Art Unit 2124
May 9, 2004